~~CONFIDENTIAL~~

26 September 1986

25X1

MEMORANDUM FOR:

VIA:

FROM:

SUBJECT: OS/ISSG PC Security Package Evaluation

Attached is an evaluation of OS/ISSG's security programs KOPY, FILE-KO, and DISK-KO. The DESCRIPTION is a brief interpretation of the documentation supplied with the file. The OBSERVATION information is obtained by disassembling the code and then single stepping through each instruction on the PC. The EVALUATION is based on the methods and implementations used to achieve the programs function.

The attached evaluations indicate serious discrepancies in program functions and considerations should be given prior to any distribution of these programs within the Agency and especially to any other government Agency to avoid embarrassment.

A brief summary of the evaluations shows:

1. KOPY offers no advantage over the DOS COPY function in either security or speed.

2. FILE-KO provides the ability to delete a file that cannot be recovered, but is conceptually deficient in its implementation.

3. DISK-KO formats a diskette that is susceptible to data loss and destruction of the Directory and File Allocation Tables. DISK-KO offers no security advantages over the DOS FORMAT program.

25X1

Distribution:

- addressee

25X1

~~CONFIDENTIAL~~

PROGRAM: KOPY

DESCRIPTION: KOPY can significantly reduce the use of the DOS Copy, Backu and Diskcopy programs. It provides a means to copy 'cleanly . . . character by character', vice the sector copy method used by the DOS function, a file from one disk or directory to another disk or directory. It allows the customer to switch destination diskettes as they become full while copying groups of files. If a destination file exists, the directory information (date and time) is compared with the directory information of the source file. If directory information is the same for both the source and the destination file, the destination file is not updated.

OBSERVATION: KOPY transfers data by reading in a large block of data (slightly under 64K bytes) using DOS function 3F which uses Interrupt 13 function 2 to Read Sectors. KOPY immediately writes the same block to the destination using DOS function 40 which uses Interrupt 15 function 87 to Move a Block of data. This procedure is repeated until the DOS read function returns with a result of zero bytes read.

KOPY does not update an existing destination file if the destinations date and time are newer than the source. The response received from KOPY is "No files needed to be copied".

EVALUATION: KOPY offers no security solutions to copying data files. Data is transferred in the same manner as the DOS COPY function, by large blocks and using the same DOS function calls. By reading blocks until the DOS function call returns with a result of zero bytes read indicates that the file size from the directory was used as a parameter to determine the amount of data to be read and not the End Of File (EOF) marker. The EOF marker is not always the last byte in the file. Data between the EOF marker and the last byte copied might contain residual data. Many applications that write data to a file will round up the last sector. Some word processors and text editors use the space after the EOF to store information about that file, i.e. tabs, page size, etc.

If a destination files exist, KOPY does not sanatize it before the copy is started.
The source may be smaller than the destination which would release the remaining sectors containing data to the free pool. These sectors would only be over written if needed for storing additional data. All data in the destination file should be over written and the file recreated to clear the directory information prior to the source being copied.

KOPY offers the ability to switch destination diskettes while copying groups of files, provided the customer has properly formatted diskettes. This feature is insignificant by the fact that KOPY is slower, has no verify parameter, and the destination cannot be renamed during copy, when compared to these advantages offered by the DOS COPY function.

PROGRAM: FILE-KO

DESCRIPTION: FILE-KO is an Overwrite and Deletion program designed to replace the DOS ERASE and DELETE functions. When DOS deletes a file, only the directory information is flagged as deleted and the sectors containing data are returned to the free pool and only overwritten if more disk space is needed. FILE-KO overwrites all data contained in the file then uses the standard DOS function to delete the file, leaving no residual data. This eliminates the possibility to unerase a file with utilities capable of changing the directory information and reallocating the sectors containing the data.

OBSERVATION: After invoking the program, the customer is prompted for the 'Pathname' of the file to be deleted. Although the buffer allows for a file specification up to 32 bytes in length, the program does not recognize anything other than a drive and a filename. No directory support is provided. The early DOS (pre version 2.1) method of using the File Control Block is used to parse, open, write, and delete the file.

The file is over written, 512 bytes at a time, with spaces and upon completion the directory entry for that file deleted.

EVALUATION: FILE-KO does offer a security solution to effectively delete files, but is conceptually deficient in its implementation because it was written for an outdated version of DOS. The program must to be re-written, using the newer DOS functions and a larger block for over writing, to increase speed and functionality. Prior to deleting the file, it should be renamed to remove any link between the file name and the type of data contained, and recreated to clear the directory information.

PROGRAM: DISK-KO

DESCRIPTION: DISK-KO formats a diskette, overwrites the entire diskette, then formats it again. This clearing process safeguards any confidential data that was previously recorded on the diskette. Only drive A can be used to format the diskette.

OBSERVATION: The diskette in drive A is formatted using the standard BIOS interrupt for the standard Double Sided/Double Density diskette. The first format is implemented for 44 tracks (DOS standard is 40). Some drives allow data to be written above track 40. This is usually used in special applications like copy protection. Next, the standard 40 tracks are overwritten with the ASCII space character, but only for 8 of the 9 sectors formatted per track. The diskette is formatted again for 40 tracks. The old boot record, File
Allocation Tables, and Directory are then written to the diskette.

There are two identical File Allocation Tables, each consisting of two 512 byte sectors. The first table was written correctly, but the last sector of the second table was over written by the first sector of the Directory.

EVALUATION: The concept of DISK-KO provides no security solutions to cleaning a diskette that the DOS FORMAT program does not offer. DISK-KO forces the use of drive A for formatting. No consideration is given to the type of drive that is available for that location. If drive A is a 1.2Meg High Density drive, only half of each track is over written and formatted, leaving the other half of each track with residual data. Both DOS and BIOS provide information about the type of drive in drive A. This information should be investigated before using a format that may not be consistent with the type of drive used.

The format procedure is done by the standard BIOS interrupt. This procedure does not return an error for unusable or bad tracks. No verification is done by DISK-KO to ensure each track is usable and all sectors are marked as good in the File Allocation Table. Lack of identification of unusable sectors can result in loss of data and possible destruction of the File Allocation Table resulting in the loss of all data on the diskette. All sectors must be verified and the File Allocation Table marked with any unusable sectors. This will ensure that applications will not write data to the unusable sectors.

The second File Allocation Table and the Directory are improperly written to the diskette after the last format. The corruption of these tables in most cases will lead to loss of all data on the diskette.

Diskettes can be formatted for a different number and size of sectors per track. Between each sector is a 'dead' space used for timing. If a diskette was previously formatted different from the standard 40 track, 9 sectors per track format, the 'dead' space of the new format may contain residual data. The concept of DISK-KO should be to perform multiple formats, each with a different number of sectors and size per track.